Bayesian Logistic Regression, Bayesian Generative Classification

Piyush Rai

Topics in Probabilistic Modeling and Inference (CS698X)

Jan 23, 2019

Bayesian Logistic Regression

- Recall that the likelihood model for logistic regression is Bernoulli (since $y \in \{0, 1\}$)

$$p(y|\boldsymbol{x}, \boldsymbol{w}) = \text{Bernoulli}(\sigma(\boldsymbol{w}^\top \boldsymbol{x})) = \left[\frac{\exp(\boldsymbol{w}^\top \boldsymbol{x})}{1 + \exp(\boldsymbol{w}^\top \boldsymbol{x})}\right]^y \left[\frac{1}{1 + \exp(\boldsymbol{w}^\top \boldsymbol{x})}\right]^{(1-y)} = \mu^y (1-\mu)^{1-y}$$

- Just like the Bayesian linear regression case, let's use a Gausian prior on $\boldsymbol{w}$

$$p(\boldsymbol{w}) = \mathcal{N}(0, \lambda^{-1}\mathbf{I}_D) \propto \exp(-\frac{\lambda}{2}\boldsymbol{w}^\top \boldsymbol{w})$$

- Given N observations $(\mathbf{X}, \boldsymbol{y}) = \{\boldsymbol{x}_n, y_n\}_{n=1}^N$, where $\mathbf{X}$ is $N \times D$ and $\boldsymbol{y}$ is $N \times 1$, the posterior over $\boldsymbol{w}$

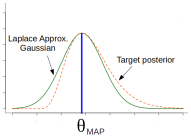
$$p(\boldsymbol{w}|\mathbf{X}, \boldsymbol{y}) = \frac{p(\boldsymbol{y}|\mathbf{X}, \boldsymbol{w})p(\boldsymbol{w})}{\int p(\boldsymbol{y}|\mathbf{X}, \boldsymbol{w})p(\boldsymbol{w})d\boldsymbol{w}} = \frac{\prod_{n=1}^N p(y_n|\boldsymbol{x}_n, \boldsymbol{w})p(\boldsymbol{w})}{\int \prod_{n=1}^N p(y_n|\boldsymbol{x}_n, \boldsymbol{w})p(\boldsymbol{w})d\boldsymbol{w}}$$

- The denominator is intractable in general (logistic-Bernoulli and Gaussian are not conjugate)
  - Can't get a closed form expression for $p(\boldsymbol{w}|\mathbf{X}, \boldsymbol{y})$. Must approximate it!
  - Several ways to do it, e.g., MCMC, variational inference, Laplace approximation (today)

Laplace Approximation of Posterior Distribution

- Approximate the posterior distribution $p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{p(\mathcal{D})} = \frac{p(\mathcal{D},\theta)}{p(\mathcal{D})}$ by the following Gaussian

$$p(\theta|\mathcal{D}) \approx \mathcal{N}(\theta_{MAP}, \mathbf{H}^{-1})$$



- Note: θ_{MAP} is the maximum-a-posteriori (MAP) estimate of θ, i.e.,

$$\theta_{MAP} = \arg\max_{\theta} p(\theta|\mathcal{D}) = \arg\max_{\theta} p(\mathcal{D},\theta) = \arg\max_{\theta} p(\mathcal{D}|\theta)p(\theta) = \arg\max_{\theta}[\log p(\mathcal{D}|\theta) + \log p(\theta)]$$

- Usually θ_{MAP} can be easily solved for (e.g., using first/second order iterative methods)
- $\mathbf{H}$ is the Hessian matrix of the negative log-posterior (or negative log-joint-prob) at θ_{MAP}

$$\mathbf{H} = -\nabla^2 \log p(\theta|\mathcal{D})\big|_{\theta=\theta_{MAP}} = -\nabla^2 \log p(\mathcal{D},\theta)\big|_{\theta=\theta_{MAP}} = -\nabla^2[\log p(\mathcal{D}|\theta) + \log p(\theta)]\big|_{\theta=\theta_{MAP}}$$

Derivation of the Laplace Approximation

- Let's write the Bayes rule as

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D},\theta)}{p(\mathcal{D})} = \frac{p(\mathcal{D},\theta)}{\int p(\mathcal{D},\theta)d\theta} = \frac{e^{\log p(\mathcal{D},\theta)}}{\int e^{\log p(\mathcal{D},\theta)}d\theta}$$

- Suppose $\log p(\mathcal{D},\theta) = f(\theta)$. Let's approximate $f(\theta)$ using its 2nd order Taylor expansion

$$f(\theta) \approx f(\theta_0) + (\theta-\theta_0)^\top \nabla f(\theta_0) + \frac{1}{2}(\theta-\theta_0)^\top \nabla^2 f(\theta_0)(\theta-\theta_0)$$

  where θ_0 is some arbitrarily chosen point in the domain of f

- Let's choose $\theta_0 = \theta_{MAP}$. Note that $\nabla f(\theta_{MAP}) = \nabla \log p(\mathcal{D},\theta_{MAP}) = 0$. Therefore

$$\log p(\mathcal{D},\theta) \approx \log p(\mathcal{D},\theta_{MAP}) + \frac{1}{2}(\theta-\theta_{MAP})^\top \nabla^2 \log p(\mathcal{D},\theta_{MAP})(\theta-\theta_{MAP})$$

Derivation of the Laplace Approximation

- Plugging in this 2nd order Taylor approximation for $\log p(\mathcal{D}, \theta)$, we have

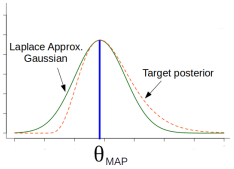
$$p(\theta|\mathcal{D}) = \frac{e^{\log p(\mathcal{D},\theta)}}{\int e^{\log p(\mathcal{D},\theta)} d\theta} \approx \frac{e^{\log p(\mathcal{D},\theta_{MAP}) + \frac{1}{2}(\theta-\theta_{MAP})^\top \nabla^2 \log p(\mathcal{D},\theta_{MAP})(\theta-\theta_{MAP})}}{\int e^{\log p(\mathcal{D},\theta_{MAP}) + \frac{1}{2}(\theta-\theta_{MAP})^\top \nabla^2 \log p(\mathcal{D},\theta_{MAP})(\theta-\theta_{MAP})} d\theta}$$

- Further simplifying, we have

$$p(\theta|\mathcal{D}) \approx \frac{e^{-\frac{1}{2}(\theta-\theta_{MAP})^\top \{-\nabla^2 \log p(\mathcal{D},\theta_{MAP})\}(\theta-\theta_{MAP})}}{\int e^{-\frac{1}{2}(\theta-\theta_{MAP})^\top \{-\nabla^2 \log p(\mathcal{D},\theta_{MAP})\}(\theta-\theta_{MAP})} d\theta}$$

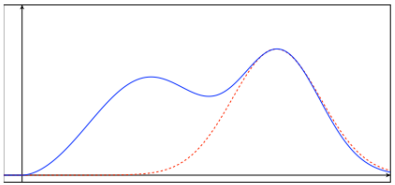
- Therefore the Laplace approximation of the posterior $p(\theta|\mathcal{D})$ is a Gaussian and is given by

$$\boxed{p(\theta|\mathcal{D}) \approx \mathcal{N}(\theta|\theta_{MAP}, \mathbf{H}^{-1})} \quad \text{where} \quad \mathbf{H} = -\nabla^2 \log p(\mathcal{D}, \theta_{MAP})$$



Properties of Laplace Approximation

- Usually straightforward if derivatives (first and second) can be computed easily
- Expensive if the number of parameters is very large (due to Hessian computation and inversion)
- Can do badly if the (true) posterior is multimodal



- Can actually apply it when working with any regularized loss function (not just probabilistic models) to get a Gaussian posterior distribution over the parameters
  - negative log-likelihood (NLL) = loss function, negative log-prior = regularizer
  - Easy exercise: Try doing this for ℓ_2 regularized least squares regression (will get the same posterior as in Bayesian linear regression)

Laplace Approximation for Bayesian Logistic Regression

- Data $\mathcal{D} = (\mathbf{X}, \boldsymbol{y})$ and parameter $\theta = \boldsymbol{w}$. The Laplace approximation of posterior will be

$$p(\boldsymbol{w}|\mathbf{X}, \boldsymbol{y}) \approx \mathcal{N}(\boldsymbol{w}_{MAP}, \mathbf{H}^{-1})$$

- The required quantities are defined as

$$\begin{aligned} \boldsymbol{w}_{MAP} &= \arg\max_{\boldsymbol{w}} \log p(\boldsymbol{w}|\boldsymbol{y}, \mathbf{X}) = \arg\max_{\boldsymbol{w}} \log p(\boldsymbol{y}, \boldsymbol{w}|\mathbf{X}) = \arg\min_{\boldsymbol{w}} [-\log p(\boldsymbol{y}, \boldsymbol{w}|\mathbf{X})] \\ \mathbf{H} &= \nabla^2 [-\log p(\boldsymbol{y}, \boldsymbol{w}|\mathbf{X})]\big|_{\boldsymbol{w}=\boldsymbol{w}_{MAP}} \end{aligned}$$

- We can compute $\boldsymbol{w}_{MAP}$ using iterative methods (gradient descent):
  - First-order (gradient) methods: $\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta \boldsymbol{g}_t$. Requires gradient $\boldsymbol{g}$ of $-\log p(\boldsymbol{y}, \boldsymbol{w}|\mathbf{X})$

$$\boldsymbol{g} = \nabla [-\log p(\boldsymbol{y}, \boldsymbol{w}|\mathbf{X})]$$

  - Second-order methods. $\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \mathbf{H}_t^{-1} \boldsymbol{g}_t$. Requires both gradient and Hessian (defined above)
- Note: When using second order methods for estimating $\boldsymbol{w}_{MAP}$, we anyway get the Hessian needed for the Laplace approximation of the posterior

An Aside: Gradient and Hessian for Logistic Regression

- The LR objective function $-\log p(\boldsymbol{y}, \boldsymbol{w}|\mathbf{X}) = -\log p(\boldsymbol{y}|\mathbf{X}, \boldsymbol{w}) - \log p(\boldsymbol{w})$ can be written as

$$-\log \prod_{n=1}^{N} p(y_n|\boldsymbol{x}_n, \boldsymbol{w}) - \log p(\boldsymbol{w}) = -\sum_{n=1}^{N} \log p(y_n|\boldsymbol{x}_n, \boldsymbol{w}) - \log p(\boldsymbol{w})$$

- For the logistic regression model, $p(y_n|\boldsymbol{x}_n, \boldsymbol{w}) = \mu_n^{y_n}(1-\mu_n)^{1-y_n}$ where $\mu_n = \frac{\exp(\boldsymbol{w}^\top \boldsymbol{x}_n)}{1+\exp(\boldsymbol{w}^\top \boldsymbol{x}_n)}$
- With a Gaussian prior $p(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}|0, \lambda^{-1}\mathbf{I}) \propto \exp(-\lambda \boldsymbol{w}^\top \boldsymbol{w})$, the gradient and Hessian will be

$$\begin{aligned} \boldsymbol{g} &= -\sum_{n=1}^{N}(y_n - \mu_n)\boldsymbol{x}_n + \lambda \mathbf{I}\boldsymbol{w} = \mathbf{X}^\top(\boldsymbol{\mu} - \boldsymbol{y}) + \lambda \boldsymbol{w} \qquad \text{(a } D \times 1 \text{ vector)} \\ \mathbf{H} &= \sum_{n=1}^{N} \mu_n(1-\mu_n)\boldsymbol{x}_n \boldsymbol{x}_n^\top + \lambda \mathbf{I} = \mathbf{X}^\top \mathbf{S}\mathbf{X} + \lambda \mathbf{I} \qquad \text{(a } D \times D \text{ matrix)} \end{aligned}$$

- $\boldsymbol{\mu} = [\mu_1, \ldots, \mu_N]^\top$ is $N \times 1$ and $\mathbf{S}$ is a $N \times N$ diagonal matrix with $S_{nn} = \mu_n(1-\mu_n)$

Logistic Regression: Predictive Distributions

- When using MLE, the predictive distribution will be

$$\begin{aligned} p(y_* = 1|\boldsymbol{x}_*, \boldsymbol{w}_{MLE}) &= \sigma(\boldsymbol{w}_{MLE}^\top \boldsymbol{x}_*) \\ p(y_*|\boldsymbol{x}_*, \boldsymbol{w}_{MLE}) &= \text{Bernoulli}(\sigma(\boldsymbol{w}_{MLE}^\top \boldsymbol{x}_*)) \end{aligned}$$

- When using MAP, the predictive distribution will be

$$\begin{aligned} p(y_* = 1|\boldsymbol{x}_*, \boldsymbol{w}_{MAP}) &= \sigma(\boldsymbol{w}_{MAP}^\top \boldsymbol{x}_*) \\ p(y_*|\boldsymbol{x}_*, \boldsymbol{w}_{MAP}) &= \text{Bernoulli}(\sigma(\boldsymbol{w}_{MAP}^\top \boldsymbol{x}_*)) \end{aligned}$$

- When using Bayesian inference, the posterior predictive distribution, based on posterior averaging

$$p(y_* = 1|\boldsymbol{x}_*, \mathbf{X}, \boldsymbol{y}) = \int p(y_* = 1|\boldsymbol{x}_*, \boldsymbol{w}) p(\boldsymbol{w}|\mathbf{X}, \boldsymbol{y}) d\boldsymbol{w} = \int \sigma(\boldsymbol{w}^\top \boldsymbol{x}_*) p(\boldsymbol{w}|\mathbf{X}, \boldsymbol{y}) d\boldsymbol{w}$$

- Above is hard in general. :-(If using the Laplace approximation for $p(\boldsymbol{w}|\mathbf{X}, \boldsymbol{y})$, it will be

$$p(y_* = 1|\boldsymbol{x}_*, \mathbf{X}, \boldsymbol{y}) \approx \int \sigma(\boldsymbol{w}^\top \boldsymbol{x}_*) \mathcal{N}(\boldsymbol{w}|\boldsymbol{w}_{MAP}, \mathbf{H}^{-1}) d\boldsymbol{w}$$

- Even after Laplace approximation for $p(\boldsymbol{w}|\mathbf{X}, \boldsymbol{y})$, the above integral to compute posterior predictive is intractable. So we will need to also approximate the predictive posterior. :-)

Posterior Predictive via Monte-Carlo Sampling

- The posterior predictive is given by the following integral

$$p(y_* = 1|\boldsymbol{x}_*, \mathbf{X}, \boldsymbol{y}) = \int \sigma(\boldsymbol{w}^\top \boldsymbol{x}_*)\mathcal{N}(\boldsymbol{w}|\boldsymbol{w}_{MAP}, \mathbf{H}^{-1})d\boldsymbol{w}$$

- Monte-Carlo approximation: Draw several samples of $\boldsymbol{w}$ from $\mathcal{N}(\boldsymbol{w}|\boldsymbol{w}_{MAP}, \mathbf{H}^{-1})$ and replace the above integral by an empirical average of $\sigma(\boldsymbol{w}^\top \boldsymbol{x}_*)$ computed using each of those samples

$$p(y_* = 1|\boldsymbol{x}_*, \mathbf{X}, \boldsymbol{y}) \approx \frac{1}{S}\sum_{s=1}^{S} \sigma(\boldsymbol{w}_s^\top \boldsymbol{x}_*)$$

  where $\boldsymbol{w}_s \sim \mathcal{N}(\boldsymbol{w}|\boldsymbol{w}_{MAP}, \mathbf{H}^{-1})$, $s = 1, \ldots, S$

- More on Monte-Carlo methods when we discuss MCMC sampling

Predictive Posterior via Probit Approximation

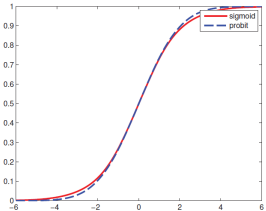
- The posterior predictive we wanted to compute was

$$p(y_* = 1|\boldsymbol{x}_*, \mathbf{X}, \boldsymbol{y}) \quad \approx \quad \int \sigma(\boldsymbol{w}^\top \boldsymbol{x}_*) \mathcal{N}(\boldsymbol{w}|\boldsymbol{w}_{MAP}, \mathbf{H}^{-1}) d\boldsymbol{w}$$

- In the above, let's replace the sigmoid $\sigma(\boldsymbol{w}^\top \boldsymbol{x}_*)$ by $\Phi(\boldsymbol{w}^\top \boldsymbol{x}_*)$, i.e., CDF of standard normal

$$\Phi(z) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z} e^{-t^2} dt \qquad \text{(Note: } z \text{ is a scalar and } 0 \leq \Phi(z) \leq 1)$$

- Note: $\Phi(z)$ is also called the probit function



- This approach relies on numerical approximation (as we will see)

Predictive Posterior via Probit Approximation

- With this approximation, the predictive posterior will be

$$p(y_* = 1|\boldsymbol{x}_*, \mathbf{X}, \boldsymbol{y}) = \int \Phi(\boldsymbol{w}^\top \boldsymbol{x}_*)\mathcal{N}(\boldsymbol{w}|\boldsymbol{w}_{MAP}, \mathbf{H}^{-1})d\boldsymbol{w} \quad \text{(an expectation)}$$

$$= \int_{-\infty}^{\infty} \Phi(a)p(a|\mu_a, \sigma_a^2)da \quad \text{(an equivalent expectation)}$$

- Since $a = \boldsymbol{w}^\top \boldsymbol{x}_* = \boldsymbol{x}_*^\top \boldsymbol{w}$, and $\boldsymbol{w}$ is normally distributed, $p(a|\mu_a, \sigma_a^2) = \mathcal{N}(a|\mu_a, \sigma_a^2)$, with $\mu_a = \boldsymbol{w}_{MAP}^\top \boldsymbol{x}_*$ and $\sigma_a^2 = \boldsymbol{x}_*^\top \mathbf{H}^{-1} \boldsymbol{x}_*$ (follows from the linear trans. property of random vars)
- Given $\mu_a = \boldsymbol{w}_{MAP}^\top \boldsymbol{x}_*$ and $\sigma_a^2 = \boldsymbol{x}_*^\top \mathbf{H}^{-1} \boldsymbol{x}_*$, the predictive posterior will be

$$p(y_* = 1|\boldsymbol{x}_*, \mathbf{X}, \boldsymbol{y}) \approx \int_{-\infty}^{\infty} \Phi(a)\mathcal{N}(a|\mu_a, \sigma_a^2)da = \Phi\left(\frac{\mu_a}{\sqrt{1+\sigma_a^2}}\right)$$

- Note that the variance σ_a^2 also "moderates" the probability of y_n being 1 (MAP would give $\Phi(\mu_a)$)
- Since logistic and probit aren't exactly identical, we usually scale a by a scalar t s.t. $t^2 = \pi/8$

$$p(y_* = 1|\boldsymbol{x}_*, \mathbf{X}, \boldsymbol{y}) = \int_{-\infty}^{\infty} \Phi(ta)\mathcal{N}(a|\mu_a, \sigma_a^2)da = \Phi\left(\frac{\mu_a}{\sqrt{t^{-2}+\sigma_a^2}}\right)$$

Bayesian Logistic Regression: Posterior over Linear Classifiers!

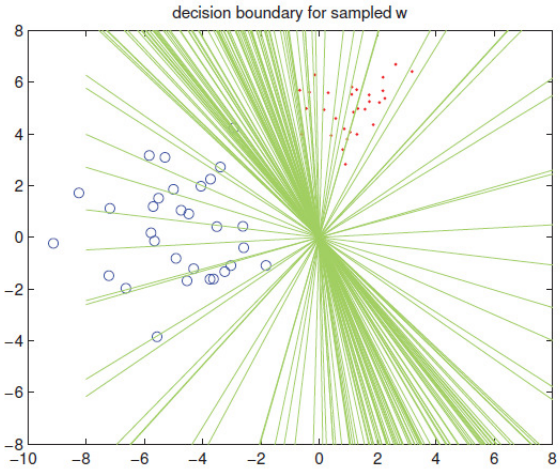
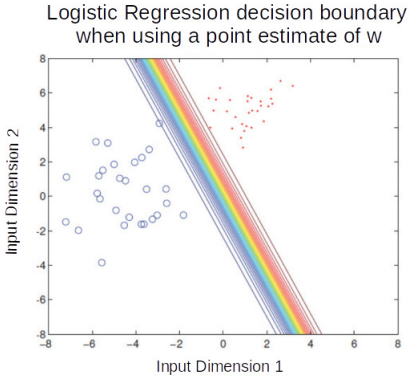
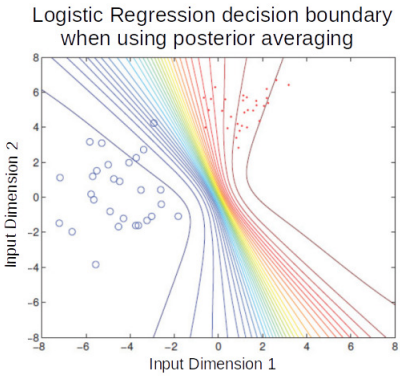


Figure courtesy: MLAPP (Murphy)

Logistic Regression: Plug-in Prediction vs Bayesian Averaging

- (Left) Predictive distribution when using a point estimate uses only a single linear hyperplane $\boldsymbol{w}$
- (Right) Posterior predictive distribution averages over many linear hyperplanes $\boldsymbol{w}$





Some Comments

- We saw basic logistic regression model and some ways to perform Bayesian inference for this model
  - We assumed the hyperparameters (e.g., precision/variance of $p(\boldsymbol{w}) = \mathcal{N}(0, \lambda^{-1}\mathbf{I})$) to be fixed. However, these can also be learned if desired
  - LR is a linear classification model. Can be extended to nonlinear classification (more on this later)
- Logistic Regression (and its Bayesian version) is widely used in probabilistic classification
- Its multiclass extension is softmax regression (which again can be treated in a Bayesian manner)
- LR and softmax some of the simplest models for discriminative classification but non-conjugate
- The Laplace approximation is one of the simplest approximations to handle non-conjugacy
- A variety of other approximate inference algorithms exist for these models
  - We will revisit LR when discussing such approximate inference methods

Bayesian Generative Classification

A Generative Model for Classification

- Consider N labeled examples $\{(\boldsymbol{x}_i, y_i)\}_{n=1}^{N}$. Assume binary labels, i.e., $y_i \in \{0, 1\}$
- Goal: Classify a new example $\boldsymbol{x}$ by assigning a label $y \in \{0, 1\}$ to it
- We will assume a Generative Model for both labels y and and features $\boldsymbol{x}$
  - What it means: We will have (probabilistic) observation models for both y as well as $\boldsymbol{x}$
  - In contrast, in Bayesian linear regression model (and Bayesian logistic regression model), we didn't model $\boldsymbol{x}$ (there, we simply conditioned y on $\boldsymbol{x}$, treating $\boldsymbol{x}$ as "fixed")
  - When we don't model $\boldsymbol{x}$ and simply model y as a function of $\boldsymbol{x}$: Discriminative Model
- Generative classification models have many benefits. E.g.,
  - Can also utilize unlabeled examples (semi-supervised learning)
  - Can handle missing/corrupted features in $\boldsymbol{x}$
  - Can properly handle cases when features in $\boldsymbol{x}$ could be of mixed type (e.g., real, binary, count)
  - And many others (more on this later during the semester)

Generative Classification: The Generative Story

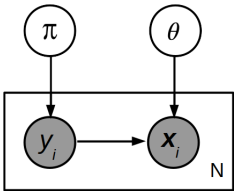
- Basic idea: Each $\boldsymbol{x}_i$ is assumed generated conditioned on the value of corresponding label y_i
- The associated generative story is as follows
  - First draw ("generate") a binary label $y_i \in \{0, 1\}$

$$y_i \sim \text{Bernoulli}(\pi)$$

  - Now draw ("generate") the feature vector $\boldsymbol{x}$ from a distribution specific to the value y_i takes

$$\boldsymbol{x}_i | y_i \sim p(\boldsymbol{x} | \theta_{y_i})$$

- The above generative model shown in "plate notation" (shaded = observed)



A Generative Model for Classification

- Our generative model for classification is

$$y_i \sim \text{Bernoulli}(\pi), \qquad \boldsymbol{x}_i | y_i \sim p(\boldsymbol{x}|\theta_{y_i})$$

- Note: We have two distributions $p(\boldsymbol{x}|\theta_0)$ and $p(\boldsymbol{x}|\theta_1)$ for feature vector $\boldsymbol{x}$ (depending on its label)
- These distributions are also known as "class-conditional distributions"
- For now, we will not assume any specific form for the distriutions $p(\boldsymbol{x}|\theta_0)$ and $p(\boldsymbol{x}|\theta_1)$
  - Depends on nature of $\boldsymbol{x}$ (real-valued vectors? binary vectors? count vectors?)
- Model parameters to be learned here: $(\pi, \theta_0, \theta_1)$
- Note: Can extend to more than 2 classes (e.g., by replacing the Bernoulli on y by multinoulli)
- Note: When y_i for each $\boldsymbol{x}_i$ is a hidden variable, we can think of it as the cluster id of $\boldsymbol{x}$
  - It then becomes a mixture model based data clustering problem (unsupervised learning)

Predicting Labels in Generative Classification

- Note: The generative model only defines $p(y|\pi)$ and $p(\boldsymbol{x}|\theta_y)$. Doesn't define $p(y|\boldsymbol{x})$
- We combine these using Bayes rule to get $p(y|\boldsymbol{x})$

$$p(y|\boldsymbol{x}) = \frac{p(y|\pi)p(\boldsymbol{x}|\theta_y)}{p(\boldsymbol{x})} = \frac{p(y|\pi)p(\boldsymbol{x}|\theta_y)}{\sum_y p(y|\pi)p(\boldsymbol{x}|\theta_y)}$$

- Parameters of distributions $p(y|\pi)$ and $p(\boldsymbol{x}|\theta_y)$ are estimated from training data using point estimation methods (MLE or MAP) or using fully Bayesian inference (discussed today)
- Once these parameters π and θ_y are estimated (point estimates, or full posterior if doing Bayesian inference), the above Bayes rule can be applied to a new input $\hat{\boldsymbol{x}}$ to compute $p(\hat{y}|\hat{\boldsymbol{x}})$
- Let's now set up the parameter estimation for π and θ_y as a Bayesian inference problem
  - Note: As we will see in the end, in this approach, computing $p(\hat{y}|\hat{\boldsymbol{x}})$ for a new input $\hat{\boldsymbol{x}}$ will NOT use a point estimate of the parameters π, θ_y but would use posterior averaging

The Priors

- Let us focus on the supervised, binary classification setting for now
- In this case, we have three parameters to be learned: π, θ_0, and θ_1
  - Probability $\pi \in (0, 1)$ of the Bernoulli. Can assume the following Beta prior

$$\pi \sim \text{Beta}(a, b)$$

  - Parameters θ_0, and θ_1 of the class-conditional distributions. Will assume the same prior on both

$$\theta_0, \theta_1 \sim p(\theta)$$

- Note: The actual form of $p(\theta)$ will depend on what the class conditional distributions $p(\boldsymbol{x}|\theta_0)$ and $p(\boldsymbol{x}|\theta_1)$ are (e.g., if these are Gaussians and if we want to learn both mean and covariance matrix of these Gaussians, then $p(\theta)$ will be some distribution over mean and covariance matrix, e.g., a Normal-inverse Wishart distribution)
- We will jointly denote the prior on π, θ_0, and θ_1 as $p(\pi, \theta_0, \theta_1) = p(\pi)p(\theta_0)p(\theta_1)$

The Likelihood

- Denote the $N \times D$ feature matrix by X and the $N \times 1$ label vector by $\boldsymbol{y}$
- Since both X and $\boldsymbol{y}$ are being modeled here, the likelihood function will be

$$\begin{aligned} p(X, \vec{y}|\pi, \theta_1, \theta_0) &= \prod_{i=1}^{N} p(x_i, y_i|\pi, \theta_1, \theta_0) \\ &= \prod_{i=1}^{N} p(x_i|y_i, \pi, \theta_1, \theta_0) p(y_i|\pi, \theta_1, \theta_0) \\ &= \prod_{i=1}^{N} p(x_i|\theta_{y_i}) p(y_i|\pi) \end{aligned}$$

The Posterior

- We need to infer the following posterior distribution

$$p(\pi, \theta_1, \theta_0 | \vec{y}, X) = \frac{p(X, \vec{y} | \pi, \theta_1, \theta_0) p(\pi, \theta_1, \theta_0)}{\int_{\Omega_\theta} \int_{\Omega_\theta} \int_0^1 p(X, \vec{y} | \pi, \theta_1, \theta_0) p(\pi, \theta_1, \theta_0) d\pi d\theta_1 d\theta_0}$$

- Note: Ω_θ denotes the domain of θ
- Might look scary at first but it isn't actually
- Recall the prior $p(\pi, \theta_0, \theta_1) = p(\pi)p(\theta_0)p(\theta_1)$. The likelihood also factorized over data points, i.e.,

$$p(X, \boldsymbol{y} | \pi, \theta_1, \theta_0) = \prod_{i=1}^{N} p(x_i | \theta_{y_i}) p(y_i | \pi)$$

- Thus, the posterior will be

$$p(\pi, \theta_1, \theta_0 | \vec{y}, X) \propto \left[\prod_{i:y_i=1} p(x_i | \theta_1) p(\theta_1) \right] \left[\prod_{i:y_i=0} p(x_i | \theta_0) p(\theta_0) \right] \left[\prod_{i=1}^{N} p(y_i | \pi) p(\pi) \right]$$

- But what about the normalization constant in the denominator?

The Posterior

- Luckily, in this case, the same factorization structure simplies the denominator as well

$$p(\pi, \theta_1, \theta_0 | \vec{y}, X) = \frac{\prod_{i:y_i=1} p(x_i|\theta_1)p(\theta_1)}{\int \prod_{i:y_i=1} p(x_i|\theta_1)p(\theta_1)d\theta_1} \cdot \frac{\prod_{i:y_i=0} p(x_i|\theta_0)p(\theta_0)}{\int \prod_{i:y_i=0} p(x_i|\theta_0)p(\theta_0)d\theta_0} \cdot \frac{\prod_{i=1}^{N} p(y_i|\pi)p(\pi)}{\int \prod_{i=1}^{N} p(y_i|\pi)p(\pi)d\pi}$$

- The above is just a product of three posterior distributions !

$$p(\pi, \theta_1, \theta_0 | \vec{y}, X) = p(\theta_1 | \{x_i : y_i = 1\}) p(\theta_0 | \{x_i : y_i = 0\}) p(\pi | \vec{y})$$

- We also know what $p(\pi|\boldsymbol{y})$ will be (recall the coin-toss example)

$$p(\pi|\vec{y}) \propto \prod_{i=1}^{N} p(y_i|\pi)p(\pi) \quad \longrightarrow \quad p(\pi|\vec{y}) = \text{Beta}(a + \textstyle\sum_i y_i, b + N - \sum_i y_i)$$

- Form of posteriors on θ_1 and θ_2 will depend on $p(\boldsymbol{x}|\theta_1)$ and $p(\theta_1)$, and $p(\boldsymbol{x}|\theta_0)$ and $p(\theta_0)$, resp.

The Predictive Posterior Distribution

- We have already seen how to compute the parameter posterior $p(\pi, \theta_1, \theta_0 | \boldsymbol{y}, X)$ for this model
- Original goal is classification. We thus also want the predictive posterior for label of a new input, i.e., $p(\hat{y}|\hat{\boldsymbol{x}})$, for which the more "complete" notation in this Bayesian setting would be $p(\hat{y}|\hat{\boldsymbol{x}}, X, \boldsymbol{y})$

$$p(\hat{y}|\hat{x}, X, \vec{y}) = \int_{\Omega_\theta} \int_{\Omega_\theta} \int_0^1 p(\hat{y}|\hat{x}, \theta_1, \theta_0, \pi) p(\theta_1, \theta_0, \pi | X, \vec{y}) d\pi d\theta_1 d\theta_0$$

- Luckily, in this case, this too has a rather simple form. Using Bayes rule, we have

$$\begin{aligned} p(\hat{y}|\hat{x}, X, \vec{y}) &= \frac{p(\hat{x}|\hat{y}, X, \vec{y}) p(\hat{y}|X, \vec{y})}{p(\hat{x}|\hat{y}=1, X, \vec{y}) p(\hat{y}=1|X, \vec{y}) + p(\hat{x}|\hat{y}=0, X, \vec{y}) p(\hat{y}=0|X, \vec{y})} \\ &= \frac{p(\hat{x}|\hat{y}, X, \vec{y}) p(\hat{y}|\vec{y})}{p(\hat{x}|\hat{y}=1, X, \vec{y}) p(\hat{y}=1|\vec{y}) + p(\hat{x}|\hat{y}=0, X, \vec{y}) p(\hat{y}=0|\vec{y})} \end{aligned}$$

- In order to compute this, we need $p(\hat{x}|\hat{y}, X, \boldsymbol{y})$ and $p(\hat{y}|\boldsymbol{y})$
  - $p(\hat{x}|\hat{y}, X, \boldsymbol{y})$: Marginal class-conditional distribution of the new input vector $\hat{x}$
  - $p(\hat{y}|\boldsymbol{y})$: Marginal probability of its label $\hat{y}$ given the labels of training data

The Predictive Posterior Distribution (Contd.)

- Predictive posterior requires computing $p(\hat{x}|\hat{y}, X, \boldsymbol{y})$ and $p(\hat{y}|\boldsymbol{y})$
- The marginal likelihood $p(\hat{x}|\hat{y}, X, \boldsymbol{y})$ of $\hat{x}$ can be computed as

$$\begin{aligned} p(\hat{x}|\hat{y}, X, \vec{y}) &= \int_{\Omega_\theta}\int_{\Omega_\theta} p(\hat{x}|\hat{y}, \theta_1, \theta_0)p(\theta_1, \theta_0|X, \vec{y})d\theta_1 d\theta_0 \\ &= \int_{\Omega_\theta} p(\hat{x}|\theta_{\hat{y}})p(\theta_{\hat{y}}|\{x_i : y_i = \hat{y}\})d\theta_{\hat{y}} \end{aligned}$$

- The above is simply the posterior predictive distribution of class $\hat{y}$. The final expression will depend on the forms of $p(\hat{x}|\theta_{\hat{y}})$ and $p(\theta_{\hat{y}}|.)$. If exp-family, we will have closed form expression!
- The marginal likelihood $p(\hat{y}|\boldsymbol{y})$ is something we have already seen (recall Bernoulli coin-toss)

$$p(\hat{y} = 1|\boldsymbol{y}) = \int p(\hat{y} = 1|\pi)p(\pi|\boldsymbol{y})d\pi = \int \pi p(\pi|\boldsymbol{y})d\pi = \frac{a + \sum_{i=1}^{N} y_i}{a + b + N}$$

- .. and $p(\hat{y} = 0|\boldsymbol{y}) = 1 - p(\hat{y} = 1|\boldsymbol{y}) = \frac{b+N-\sum_{i=1}^{N} y_i}{a+b+N}$

A Simple/Special Case: Naïve Bayes Assumption

- Usually the most critical choice in generative classification is that of class conditional $p(\boldsymbol{x}|\theta_y)$
- Very complex $p(\boldsymbol{x}|\theta_y)$ with lots of parameters may make estimation difficult
- Often however we can choose simple forms of $p(\boldsymbol{x}|\theta_y)$ to make estimation easier
- The naïve Bayes assumption: The conditional distribution $p(\boldsymbol{x}|\theta_y)$ factorizes over individual features (or over groups of features)
  - Suppose the features of $\hat{x}$ can be partitioned into v groups $\hat{x} = \{\hat{x}(j)\}_{j=1}^{v}$
  - Can also assume a similar partitioning for the parameters $\theta_{\hat{y}}$
  - This further simplifies calculation of marginal likelihood $p(\hat{x}|\hat{y}, X, \boldsymbol{y})$

$$
\begin{aligned}
p(\hat{x}|\hat{y}, X, \vec{y}) &= \int_{\Omega_\theta} \prod_{j=1}^{v} p(\hat{x}(j)|\theta_{\hat{y}}(j)) p(\theta_{\hat{y}}(j)|\{x_i(j) : y_i = \hat{y}\}) d\theta_{\hat{y}} \\
&= \prod_{j=1}^{v} \int p(\hat{x}(j)|\theta_{\hat{y}}(j)) p(\theta_{\hat{y}}(j)|\{x_i(j) : y_i = \hat{y}\}) d\theta_{\hat{y}}(j)
\end{aligned}
$$

  - This modeling choice in a Bayesian setting gives rise to a "Bayesian naïve Bayes" model

A Simple/Special Case: Naïve Bayes Assumption

- In the Bayesian naïve Bayes model, we can still choose different types of class conditional $p(\boldsymbol{x}|\theta_y)$
  - Gaussian naïve Bayes: if $\boldsymbol{x}$ is modeled using a multivariate Gaussian (assumed factorized as per the naïve Bayes assumption)
  - Multivariate Bernoulli naïve Bayes: if $\boldsymbol{x}$ is modeled using a multivariate Bernoulli (assumed factorized as per the naïve Bayes assumption)
- MLAPP (Murphy) Section 3.5.1.2 and 3.5.5 contains an example of Multivariate Bernoulli case